Routage et agrégation de données dans les réseaux de capteurs sans fil structurés en *clusters* auto-stabilisants

Mandicou Ba ‡, Olivier Flauzac *, Ibrahima Niang†, and Florent Nolot *

‡ Université Cheikh Anta Diop (UCAD), École Supérieur Polytechnique (ESP)
Département Génie Informatique (DGI), mandicou.ba@esp.sn

* Université de Reims Champagne-Ardenne (URCA), Laboratoire CReSTIC EA 3804 - Équipe SysCom, {olivier.flauzac, florent.nolot}@univ-reims.fr

† Université Cheikh Anta Diop (UCAD), Département de Mathématiques et Informatique (DMI)
Laboratoire d'Informatique de Dakar (LID), iniang@ucad.sn

RÉSUMÉ. Dans cet article, nous menons une étude complète visant à proposer trois stratégies de routage, intégrant différents niveau d'agrégation, afin d'acheminer les données collectées dans les Réseaux de Capteurs Sans Fil (RCSF) structurés en *clusters* auto-stabilisants. Ces trois scénarios sont : *(i)* le Routage Sans Agrégation (RSA), *(ii)* le Routage avec Agrégation Partielle (RAP) et *(iii)* le Routage avec agrégation Totale (RAT). Ces derniers se fondent sur un schéma de *clustering* auto-stabilisant où est intégré un système d'agents coopératifs. Nous validons ces trois scénarios par simulation sous OMNeT++ en évaluant et comparant leurs performances en termes de délai de bout en bout, de consommation énergétique et de durée de vie du réseau. Les résultats de simulation montrent que le RSA minimise les délais de communication, le RAP réduit la consommation énergétique et le RAT prolonge la durée de vie des *cluster-heads*

ABSTRACT. In this paper, we conduct a complete study aimed to propose three routing approaches in order to transfer data in Wireless Sensors Networks (WSNs) structured into self-stabilizing clusters. These approaches are : *(i)* non-aggregation scenario, *(ii)* partially-decentralized scenario aggregation and *(iii)* fully-decentralized aggregation scenario. These are based on a self-stabilizing clustering algorithm in which is integrated an intelligent multi-agent systems. To validate the proposed approaches, we evaluate and compare its end-to-end delay, energy consumption and network lifetime. Simulation results, under OMNeT++, show that the non-aggregation aggregation scenario provides a better end-to-end delay, the partially-decentralized aggregation scenario reduces the energy consumption and prolongs the network lifetime and fully-decentralized aggregation prolongs cluster-head's lifetime.

MOTS-CLÉS : RCSF, *Clustering* auto-stabilisant, Routage, Agrégation, Multi-Agents, OMNeT++.

KEYWORDS : WSNs, Self-stabilizing Clustering, Routing, Aggregation, Multi-Agents, OMNeT++.

1. Introduction

Les Réseaux de Capteurs Sans Fil (RCSF) consistent en un grand nombre de dispositifs appelés capteurs. Ces derniers sont dotés de la capacité de collecter des grandeurs physiques telles que la température, la pression, le pH, etc. dans une zone d'étude. Puis, d'effectuer des traitements sur les données recueillies avant de coopérer entre eux pour les acheminer vers un centre de contrôle appelé station de base (*Base Station* (BS)). Du fait de la taille réduite des capteurs et leur faible de coût de production, les RCSF offrent de nombreuses applications pratiques et parfois sensibles notamment dans le domaine militaire, médical, environnemental etc. [2].

Cependant, dans les RCSF, deux contraintes majeures demeurent : *(i)* la faible ressource énergétique des capteurs et *(ii)* la vulnérabilité aux pannes. D'une part, les communications sont la principale source de consommation énergétique [40]. Pour réduire les communications et donc minimiser la consommation énergétique, une solution consiste à structurer le réseau en *clusters* [22]. D'autre part, les RCSF sont sujets à des pannes diverses : dysfonctionnement d'un des composants d'un capteur (module de captage ou de communication par exemple), modifications topologiques causées par la mobilité des nœuds, etc. [49]. Dans un système distribué comme les RCSF, Johnen et Mekhaldi ont montré dans [26] que l'auto-stabilisation une solution efficace de tolérance aux pannes.

Plusieurs approches de *clustering* auto-stabilisantes ont été proposées dans le but d'optimiser les communications et de tolérer les pannes transitoires [4, 11, 16, 27, 33, 34, 37]. Ces dernières peuvent être utilisées pour le routage de l'information dans les RCSF. En outre, dans les RCSF structurés en *clusters*, pour économiser l'énergie des capteurs, il est nécessaire de limiter les échanges de messages dans le réseau [3, 29, 40, 46]. Or, le routage de l'information depuis un nœud source jusqu'à la BS nécessite la retransmission de messages à travers le réseau. Pour réduire les messages qui transitent dans le réseau, une solution consiste à agréger les données lors de leurs acheminements [1, 18, 47]. Cependant, l'agrégation de données nécessite une attente au niveau des nœuds pour recevoir les informations venant de leurs voisins. Cette attente dure un temps prédéterminé et implique une augmentation du délai de bout en bout. En contre partie, l'agrégation permet de limiter le nombre de messages transmis et donc d'économiser l'énergie des capteurs avec comme conséquence le prolongement de la durée de vie du réseau. Elle permet également de faire un certains nombre de traitements préalables sur les données avant leur acheminement. Cependant, à notre connaissance, aucun mécanisme de routage avec agrégation n'a été proposé pour les solutions de *clustering* auto-stabilisantes décrites dans [11, 16, 27, 33, 34, 37].

Sur la base de tous les constats évoqués ci-dessus, nous menons dans ce papier une étude complète visant à proposer différentes stratégies de routage prenant compte des contraintes telles que le délai de bout en bout, la conservation énergétique ou l'acheminement de préalablement traitées. Pour ce faire, nous reprenons notre schéma de *clustering* auto-stabilisant proposé dans [4] et nous l'associons avec un système d'agents coopéra-

tifs proposé par Sardouk et *al.* [1] [42], afin de proposer trois scénarios de routage intégrant différent niveau d'agrégation. Ces trois scénarios sont : *(i)* le Routage Sans Agrégation (RSA), *(ii)* le Routage avec Agrégation Partielle (RAP) et *(iii)* le Routage avec Agrégation Totale (RAT). Les résultats de nos simulations sous OMNeT++ montrent que le RSA minimise les délais de communication, le RAP réduit la consommation énergétique totale et le RAT prolonge la durée de vie des *cluster-heads*.

Le suite de cet article est organisée comme suit. Dans la section 2, nous décrivons notre schéma de *clustering* auto-stabilisant. Ensuite, dans la section 3, nous décrivons les mécanismes adoptés pour la construction des différentes routes du réseau. La section 4 détaille le système d'agents coopératifs utilisé dans le mécanisme d'agrégation totale. Au niveau de la section 5, nous décrivons les trois scénarios de routage proposés pour l'acheminement de l'information dans un réseau de capteurs, puis, dans la section 6, nous évaluons et comparons leurs performances. Enfin, dans la section 7, nous concluons cet article et présentons quelques perspectives de recherche.

2. Structuration du réseau en *clusters* auto-stabilsants

Dans cette section, nous présentons notre solution de *clustering* sur lequel s'appuie les différentes approches de routage que nous proposons. Partant des constatations que les algorithmes de *clustering* auto-stabilisants sur un modèle à états [11, 16, 27] ne sont pas adaptés dans le contexte des RCSF et que les solutions sur un modèle à passage messages [33, 34, 37] engendrent d'importantes communications, nous avons proposé dans dans [4] un algorithme nommé SDEAC (self-Stabilizing Distributed Energy-Aware k-hops Clustering), afin d'optimiser les communications et de tolérer les pannes transitoires. Comme illustré dans la figure 1, SDEAC construit des *clusters* non-recouvrants de diamètre au plus $2k$. Il ne nécessite aucune initialisation, utilise le critère de l'identité maximale pour l'élection des *cluster-heads* et se fonde sur un modèle asynchrone à passage de messages. Partant d'une configuration quelconque et sans occurrence de pannes transitoires, SDEAC auto-organise le réseau en *clusters* en un nombre fini de transitions. La particularité de SDEAC est qu'elle utilise uniquement une connaissance du voisinage à 1 saut (informations locales) pour construire des *clusters* à k sauts contrairement aux solutions [16, 11, 34, 37] qui considèrent un voisinage à k sauts.

Dans un premier temps, nous avons validé SDEAC par preuve une formelle, en montrant qu'une configuration légale est atteinte en au plus $n + 2$ transitions [4]. De plus, il requiert $(\Delta_u + 1) * \log(2n + k + 3)$ bits pour chaque nœud u du réseau où, n la taille du réseau, k le rayon maximal des *clusters* et Δ_u le nombre de voisins de u. Le tableau 1 illustre une comparaison analytique du temps de stabilisation, de l'occupation mémoire et du voisinage de SDEAC avec ceux des solutions [16, 11, 34]. Dans un deuxième temps, nous avons mené dans [5] une étude comparative de SDEAC avec la solution de Mitton et *al.* décrite dans [37] qui opère dans un modèle identique. Les résultats ont montré que SDEAC réduit la consommation énergétique par un facteur d'au moins 2. Enfin dans un troisième temps, nous avons proposé une généralisation du critère d'élection des *cluster-*

1. Dans cet article nous fusionnons les travaux décrits dans [4, 42] qui s'inscrivent dans le cadre du projet régional CPER CapSec ROFICA co-financé par la région de Champagne-Ardenne et les FEDER.

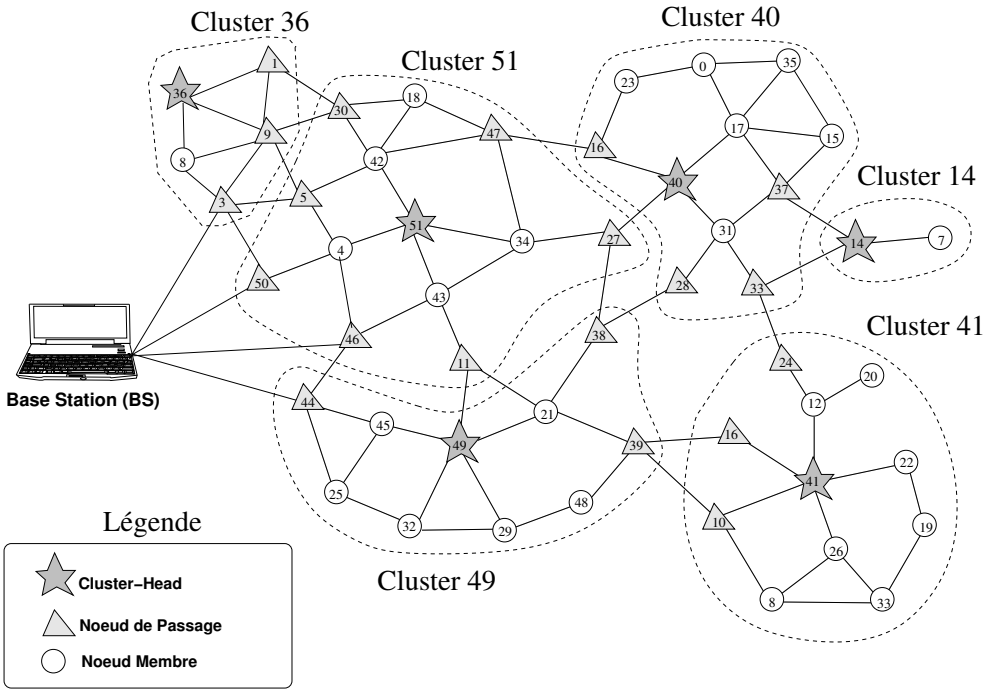


Figure 1 – SDEAC : structuration d'un réseau de capteurs en *clusters* auto-stabilisants

heads dans [6] et évalué son comportement en présence de pannes transitoires dans [7]. Les résultats reflètent que SDEAC est une solution de *clustering* générique. En plus du critère de l'identité maximale, elle peut utiliser le degré ou l'énergie résiduelle des nœuds pour élire les *cluster-heads*. De plus, suite à l'occurrence de pannes, la consommation énergétique lors du *re-clustering* est très inférieure à celle du *clustering*.

Tableau 1 – Comparaison analytique des temps de stablilsation et occupations mémoires

	Solutions	Voisinage	Temps de stabilisation	Occupation mémoire
***Clusters* de diamètre** $2k$	**SDEAC**	1 saut	$n+2$	$\log(2n+k+3)$
	Minimal [16]	k sauts	$O(n)\,;O(n^2)$	$O(\log(n))$
	Flood [15]	k sauts	$O(k)$	$O(k \times \log(n))$
	(x,k)-Clustering [34]	k sauts	$O(k)\,;O(g \times k \times \log(n))$	$O(\|G_u^k\| \times (\log(n)+\log(k)))$
	k-Clustering [11]	k+1 sauts	$O(n \times k)$	$O(log(n)+log(k))$

Dans la suite, nous considérons un réseau de capteurs sans fil structuré en *clusters*. Ensuite, nous proposons plusieurs solutions de routage, intégrant différents niveau d'agrégation de données, pour l'acheminement de l'information depuis un capteur source jusqu'à la station de base.

3. Construction des routes sur le réseau capteurs clusterisé

Dans cette section, nous présentons les mécanismes de construction des routes sur un réseau de capteur structuré en *clusters*. Ces routes sont utilisées durant les phases d'acheminement des données collectées dans le réseau.

3.1. Adaptation du protocole DSR pour la construction de routes dans un réseau de capteurs structuré en *clusters*

Pour construire les différentes routes, depuis tous les nœuds capteurs du réseau jusqu'à la BS, nous utilisons le principe du protocole standard nommé *Dynamic Source Routing* (DSR) [28]. DSR se fonde sur une technique d'inondation pour découvrir l'itinéraire d'une source vers une destination. Or, l'inondation peut être coûteuse en termes d'échanges de messages. Nous proposons donc d'adapter DSR dans nos structures en *clusters* afin de réduire les messages lors de la phase de construction des routes. L'approche que nous mettons en œuvre fonctionne en deux phases : (*1*) une phase de *découverte des routes* durant laquelle les nœuds capteurs obtiennent des informations sur leurs voisinages ainsi que leurs nœuds de passage (i.e. le premier voisin direct d'un nœud sur son chemin vers la BS) et (*2*) une phase de *mise à jour des routes* qui permet aux nœuds de maintenir à jour les informations de routage acquises lors de la phase de découverte des routes.

3.1.1. Découverte des routes

Dans le but de découvrir les routes qui mènent vers la BS tout en limitant les échanges de messages, seuls les *cluster-heads*, une fois qu'ils se considèrent stables comme nous l'avons défini dans [4] , initient la construction des routes en envoyant un message de découverte des routes vers la BS (cf. Figure 2). Ces messages vont être retransmis à destination de la BS et vont récolter dans un vecteur toutes les identités de nœuds sur le chemin. La BS maintient une table de routes contenant toutes les routes vers tous les *cluster-heads* du réseau. Dans la figure 2, la table $Route_{BS}$ illustre un exemple de quelques routes entre la BS et chaque *cluster-head* du réseau.

Une fois arrivée au niveau de la BS, les messages de découverte des routes reprennent les chemins inverses vers les *cluster-heads* initiateurs. En récupérant les messages revenus de la BS, chaque *cluster-head* du réseau maintient une table de routes contenant toutes les routes connues vers la BS. Ainsi, pour envoyer les données, les *cluster-heads* pourront par exemple choisir les routes les plus courtes. Dans la figure 2, $Routes_{49}$ illustre quelques routes du *cluster-head* 49.

Chaque nœud u qui n'est pas *cluster-head* (c'est-à-dire de statut Nœud (NM) ou Nœud de Passage (NP) voire Figure 2) connait déjà ses voisins directs et son nœud de passage vers son *cluster-head*. Ces informations sont obtenues lors de la structuration du réseau par SDEAC et sont présentes dans sa table de voisinage $StateNeigh_u$. Dans la figure 2, $StateNeigh_{47}$ donne l'exemple de la table de voisinage du nœud 47 appartenant

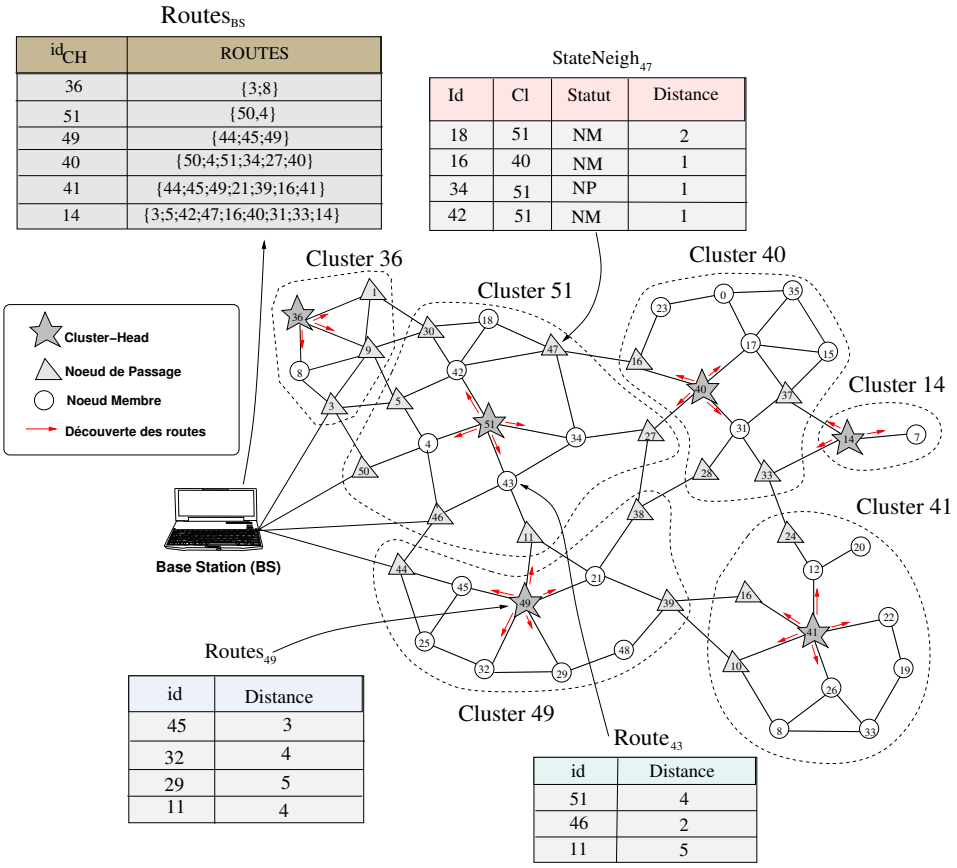


Figure 2 – Construction des routes

au **Cluster 51**. Ainsi, durant la phase de construction des routes, un nœud de statut NM ou NP recueille les informations sur les nœuds de passage qu'il peut contacter pour envoyer ses informations en direction de la BS. Ces informations sont les seules collectées par un NM ou NP lors de la construction des routes. Dans la figure 2, $Routes_{43}$ montre quelques routes du nœud 43 appartenant au **Cluster 51**.

3.1.2. Mise à jour des routes

Les nœuds mettent à jour leurs tables de routage avec une technique de communication d'informations pro-active de type *push*. Cette opération est déclenchée lorsqu'un capteur n'a plus suffisamment d'énergie pour participer à l'opération de routage. Le seuil d'énergie critique est laissé à l'appréciation de l'administrateur du réseau : une méthode consisterait à de fixer le seuil à 25% du niveau initial. Cela permet ainsi aux voisins qui le considèrent comme nœud de passage de mettre à jour leur table de routage et d'en choisir un autre pour cette fonction. Comme le décrit la figure 3, la mise à jour des routes se fait au moyen de quatre types de messages :

• Messages de prévention (*P*) : un nœud ayant un niveau de batterie qui a atteint un seuil critique prévient ses voisins afin que ceux-ci puissent éviter dans le futur de le

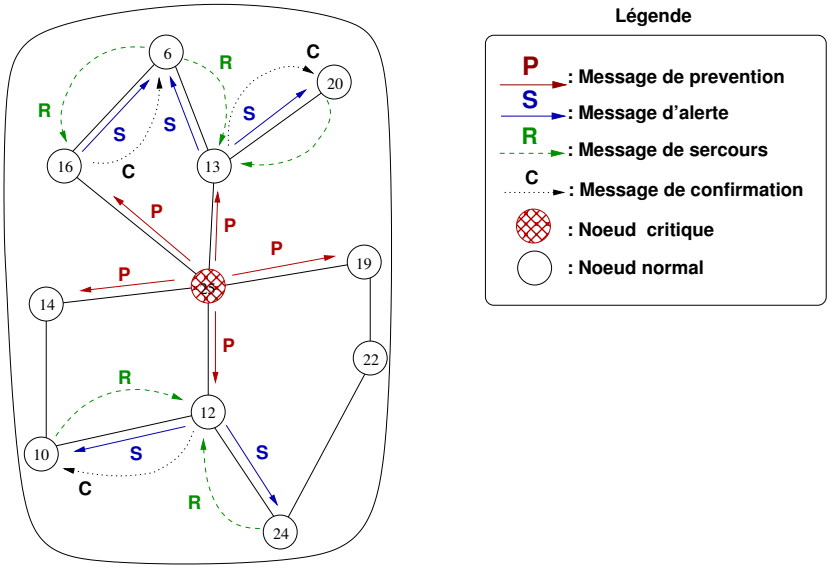


Figure 3 – Maintenance du voisinage

solliciter pour router leurs informations vers le *cluster-head* ou la station de base ;

• Messages d'aide (*S*) : un nœud qui reçoit un message de prévention et qui utilise le nœud source comme nœud de passage, diffuse un message d'aide à ses autres voisins à 1 saut pour leur demander de devenir son nœud de passage ;

• Messages d'assistance (*R*) : lorsqu'un nœud reçoit un message d'aide, il répond au demandeur en lui envoyant un message contenant son adresse et son niveau d'énergie résiduelle ;

• Messages de confirmation (*C*) : après la réception de messages d'assistance, le nœud en quête de passerelle choisit le voisin qui a la plus grande quantité d'énergie résiduelle comme passerelle puis informe celui-ci du changement en lui envoyant un message de confirmation. Lorsque le nœud passerelle reçoit ce message, il met à jour sa base de connaissance, qui contient entre autres la liste des nœuds le considérant comme passerelle.

4. Agrégation de données à base d'agents coopératifs

Après avoir présenté l'algorithme de structuration auto-stabilisant des réseaux de capteurs et les mécanismes de construction des routes, nous présentons dans cette section une approche d'agrégation de données à base d'agents coopératifs. Cette approche sera utilisé dans l'une des solutions de routage présentées à la section suivante.

En partant du fait que la communication est de loin la première source de consommation d'énergie [40], afin de réduire les messages qui transitent dans le réseau, nous proposons un traitement local des données avant leur envoi vers la station de base. Pour cela, nous appliquons les travaux de Merghem-Boulahia et *al.* [42, 41, 45, 44, 43] [2] dans le réseau que nous structurons avec SDEAC. Le but est d'arriver à une approche décen-

2. Rappelons que nos travaux ainsi que ceux de Merghem-Boulahia et *al.* s'inscrivent dans le cadre du projet CPER CapSec Rofica.

tralisée d'agrégation de données se fondant sur un système multi-agents afin d'envoyer un plus gros volume de données vers BS pour une meilleure vue sur la zone de captage et prise de décision.

4.1. Scénarios d'agrégation

Un agent implémenté sur chaque nœud permet d'utiliser les informations de routage pour établir une vue située locale et maintient ainsi une base de connaissance. Chaque agent pourra décider selon une stratégie donnée et d'une manière complètement décentralisée s'il veut coopérer ou non avec un autre agent.

Quand un capteur collecte une information sur son environnement, son agent décide si celle-ci est importante. Si c'est le cas, il initie l'agrégation en envoyant une demande de coopération à ses voisins directs afin qu'ils puissent participer s'ils le souhaitent à la session d'agrégation en cours. Les agents des nœuds voisins prennent la décision de coopérer ou non selon une stratégie décrite à la section suivante. Si un agent décide de coopérer, alors il envoie au demandeur les informations collectées par son nœud.

Si l'agent qui reçoit la demande de coopération est utilisé comme passerelle par le demandeur, alors il envoie directement une demande de coopération à ses voisins directs en attendant de recevoir l'agrégat calculé par l'agent demandeur. Lorsque le nœud demandeur reçoit les informations de ses voisins directs, il les concatène et élimine les redondances avant d'envoyer le résultat (i.e. l'agrégat) à sa passerelle. Cette dernière fusionne les informations des voisins et l'agrégat reçu, puis envoie le résultat à sa passerelle. Cette procédure se répète jusqu'à ce que les données agrégées atteignent la station de base.

4.2. Coopération entre agents

Nous définissons une stratégie de coopération en prenant en compte plusieurs paramètres dans le processus de prise de décision, tels que le niveau d'importance des informations collectées, le niveau d'énergie résiduelle des capteurs, la position des nœuds dans le réseau et le degré des nœuds. Ainsi, en fonction de la valeur de ces paramètres, chaque agent calcule un coefficient $\mathcal{R}$ qui détermine la pertinence de la coopération qui lui permet de décider s'il coopère ou non à l'opération de routage. Celui-ci est calculé selon l'équation 1. Par exemple, lorsqu'un nœud ne dispose que d'une petite quantité d'énergie, il peut refuser de participer au routage de certaines informations ou de toutes les informations lorsque son niveau de batterie est critique. Ainsi, il économise son énergie pour ne l'utiliser que dans la capture et la transmission de ses propres informations. Les paramètres de coopération utilisés sont décrits comme suit :

• Énergie résiduelle ($\mathcal{E}$) : Ce paramètre clé permet aux agents de maximiser la durée de vie de leur réseau en ne participant au routage que si la valeur de ce paramètre est assez élevée. Nous considérons le rapport entre l'énergie restante $\mathcal{E}_r^t$ à un instant t et l'énergie maximale $\mathcal{E}_i$ du capteur au moment de son déploiement, comme illustré dans cette équation : $\mathcal{E} = \frac{\mathcal{E}_r^t}{\mathcal{E}_i}$;

• Degré ($\mathcal{D}$) : Il permet en outre d'identifier les zones denses dans le réseau et celles où il y a peu de capteurs. Lorsqu'un nœud dispose d'un nombre de voisins très élevé, il consomme son énergie très rapidement et il a plus de chance de trouver un voisin avec qui coopérer qu'un autre nœud ayant un faible degré. Ses voisins évitent alors de le solliciter

souvent avec des demandes de coopération ;

• Position ($\mathcal{P}$) : Nous définissons trois positions possibles pour un nœud dans le réseau : normale, bordure ou critique. La position d'un nœud est considérée critique si celui ci relie deux parties du réseau. En outre, il représente une passerelle dans notre architecture en *clusters*. Dans ce cas, les nœuds de statut NP ou CH vont être des nœuds critiques. Un nœud est dit de bordure s'il est à "l'extrémité" d'un *cluster*. Tout autre nœud occupe une position "normale". Celui-ci est entouré par plusieurs voisins et ne représente pas une passerelle. Les nœuds normaux ou de bordures sont de statut SN. A titre d'exemple, dans la figure 4, les nœuds 46 et 50 du **Cluster 51** sont des nœuds critiques, les 19, 22 et 33 du **Cluster 41** sont des nœuds de bordures et les nœuds 4, 34, 42 et 43 du **Cluster 51** sont des nœuds normaux.

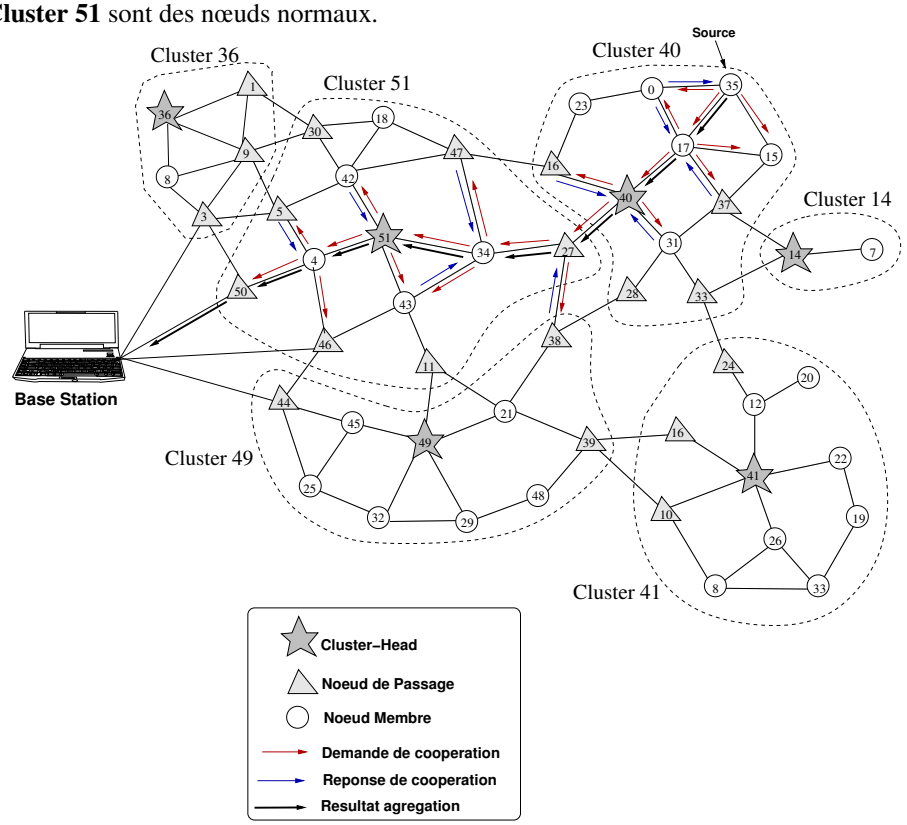


Figure 4 – Mécanisme d'agrégation de données à base d'agents coopératifs

• Importance des informations ($\mathcal{I}$) : Ce paramètre dépend surtout du type d'application cible. Une information est jugée importante dans le cas où par exemple l'écart entre la nouvelle valeur perçue et l'ancienne est assez éloigné et dépasse un certain seuil.

En fonction de la valeur de chacun de ces paramètres $\mathcal{E}$, $\mathcal{D}$, $\mathcal{P}$, $\mathcal{I}$ et de leurs poids respectifs w_e, w_d,w_p, w_i, nous calculons le coefficient de coopération selon l'équation 1.

$$\mathcal{R} = \mathcal{E} \times w_e + \frac{1}{\mathcal{D}} \times w_d + \mathcal{P} \times w_p + \mathcal{I} \times w_i \qquad [1]$$

La figure 4 illustre l'exemple du nœud source d'identité 35 du **Cluster 40** qui détecte un événement important et demande à ses voisins directs 0, 15 et 17 de coopérer. Il agrège les données reçues avec les siennes puis envoie l'agrégat qui en résulte à son nœud de passage d'identité 17. Dans cet exemple, le nœud 46 du **cluster 51** décide de ne pas coopérer étant donnée sa position critique (voisin direct de la BS et passerelle) et ses réserves d'énergie qui s'épuisent vite à cause de son emplacement.

5. Acheminement de l'information dans un réseau de capteurs clusterisé

Après avoir présenté notre approche de structuration du réseau en *clusters*, notre mécanisme de construction des routes et décrit notre scénario d'agrégation à base d'un système multi-agent coopératif, nous décrivons maintenant trois scénarios proposés pour le routage des informations collectées depuis les nœuds du réseau jusqu'à la BS. Ces trois scénarios sont : (*i*) le Routage Sans Agrégation (RSA), le (*ii*) Routage avec Agrégation Partielle (RAP) et (*iii*) le Routage avec Agrégation Totale (RAT).

5.1. Routage Sans Agrégation (RSA)

Dans le RSA, comme illustré dans la figure 5, les données collectées par les nœuds sont directement envoyées à la BS. Un nœud capteur u qui collecte une donnée l'envoie à destination de son *cluster-head* en passant par sa passerelle ($np_{(u,CH_u)}$) dans le *cluster*. Tout *cluster-head* qui reçoit un message venant de l'un des nœuds de son *cluster* le transfère directement vers la BS via sa passerelle $np_{(CH,BS)}$. Ainsi, tout nœud v du chemin transfère le message vers la BS via sa passerelle $np_{(v,BS)}$. Cette procédure se répète jusqu'à ce que le message arrive au niveau de la BS.

5.2. Routage avec Agrégation Partielle (RAP)

Pour le RAP, comme montré dans la figure 6, nous proposons un routage avec une agrégation partielle (agrégation au niveau des *cluster-heads* uniquement). Tout u dans un *cluster* qui collecte une donnée l'envoie vers son *cluster-head* via sa passerelle ($np_{(u,CH_u)}$). Ainsi, chaque *cluster-head* collecte et fusionne toutes les données émanant de son *cluster* en un seul agrégat. Puis, l'agrégat est envoyé par le *cluster-head* en direction de la BS via la passerelle $np_{(CH,BS)}$. Et tout nœud v sur le chemin retransmet à son tour l'agrégat vers la BS via sa passerelle $np_{(v,BS)}$. Cette procédure se répète jusqu'à ce que le message arrive au niveau de la BS.

5.3. Routage avec Agrégation Totale (RAT)

Pour le troisième scénario de routage, illustré dans la figure 7, nous proposons un routage avec une agrégation totale (RAP) en mettant en œuvre le mécanisme d'agrégation de données à base d'agents coopératifs décrit dans la Section 4. Comme précédemment décrit, un nœud qui capture une donnée pertinente demande à ses voisins leurs données puis fusionne partiellement toutes les données reçues du voisinage et envoie l'agrégat partiel à destination de la BS. Cette procédure se répète au niveau de chaque nœud le long du chemin jusqu'à ce que le message arrive au niveau de la BS.

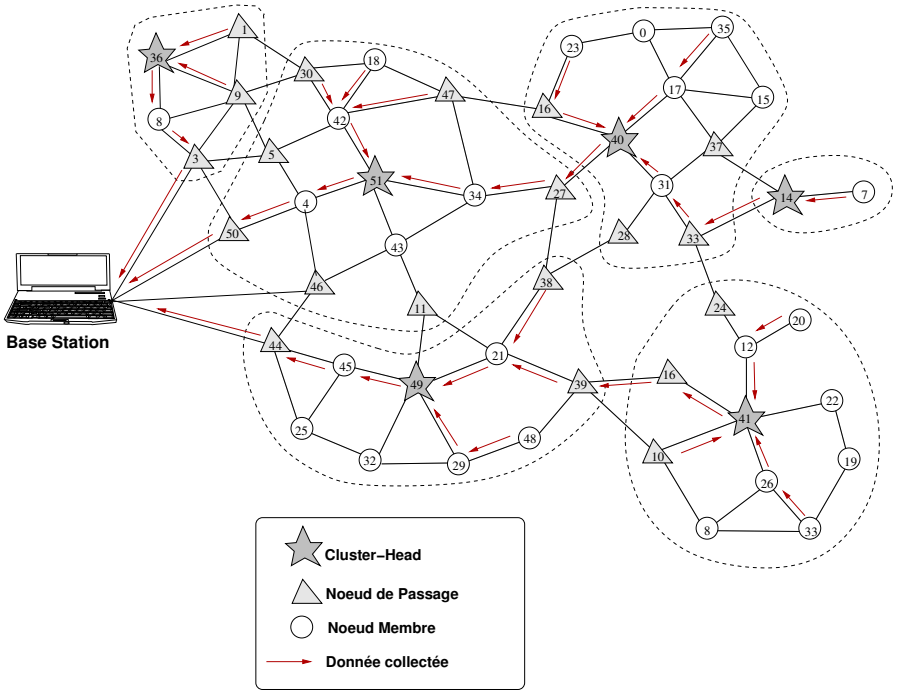


Figure 5 – Scénario de Routage Sans Agrégation (RSA)

6. Évaluation des performances de l'acheminement de l'information

Nous présentons maintenant une évaluation des performances des trois approches de routage que nous proposons pour l'acheminement de l'information depuis un capteur source jusqu'à la BS. Pour ce faire, tout comme nous avons implémenté SDEAC[4] dans OMNeT++ [3], nous implémentons le mécanisme de construction des routes ainsi que les trois scénarios de routage (RSA, RAP et RAT). Ensuite, nous comparons les coûts et performances des trois scénarios de routage selon les métriques suivantes : *(i)* délai de boue en bout, *(ii)* la consommation énergétique, *(iii)* et la durée de vie du réseau.

6.1. Métriques d'évaluation

Nous proposons d'évaluer les coûts et performances des trois scénarios de RSA, de RAP et de RAT en mesurant et comparant trois métriques : le délai de bout en bout, la consommation énergétique et la durée de vie du réseau.

6.1.1. Modèle énergétique et consommation énergétique

Dans cette section, nous présentons le modèle énergétique utilisé ainsi que la métrique de consommation énergétique.

3. http ://www.omnetpp.org

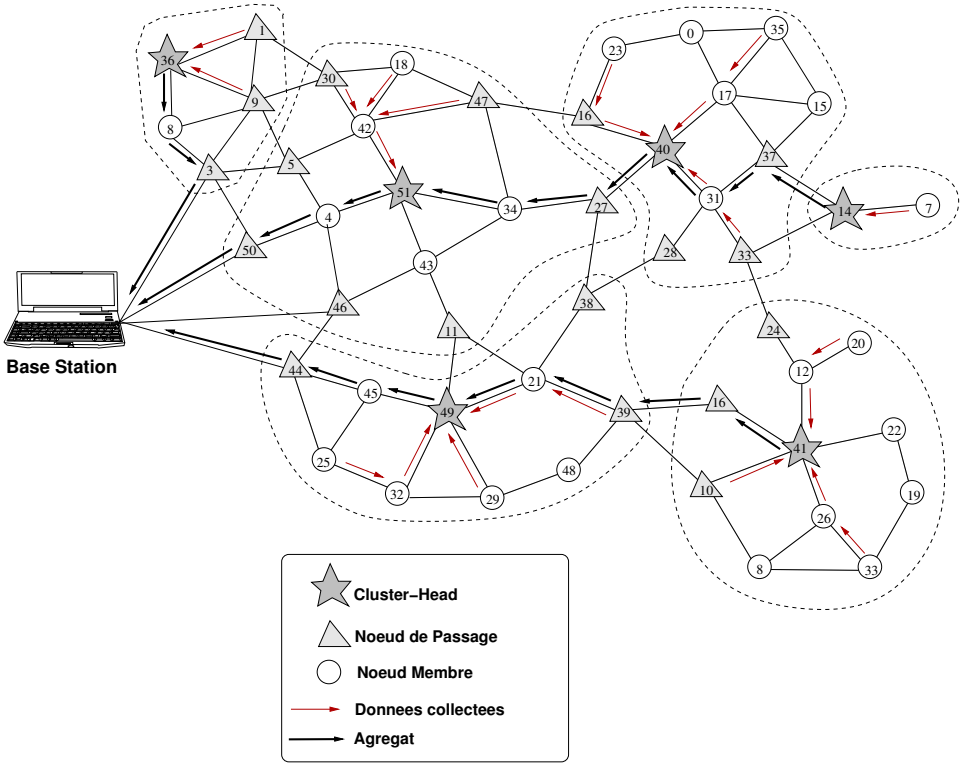


Figure 6 – Scénario de Routage avec Agrégation Partielle (RAP)

Nous utilisons le modèle énergétique de référence proposé par Heinzelman et *al.* dans [22] pour les réseaux de capteurs sans fil. Selon ce modèle, chaque nœud consomme une quantité d'énergie égale à E_{Tx} lorsqu'il transmet un message de l bits sur une distance de d mètres. Comme le montre l'équation 2, lorsque la distance dépasse un certain seuil, le modèle de consommation d'énergie change car le capteur consomme plus d'énergie.

$$E_{Tx}(l,d) = \begin{cases} l * E_{elec} + l * \varepsilon_{fs} * d^2, & \text{si } d < d_0; \\ \\ l * E_{elec} + l * \varepsilon_{mp} * d^4, & \text{si } d \geq d_0. \end{cases} \quad [2]$$

Lorsqu'un nœud reçoit un message, il consomme la quantité d'énergie E_{Rx} décrite dans l'équation 3.

$$E_{Rx}(l) = l * E_{elec} \quad [3]$$

Les paramètres des équations 2 et 3 sont donnés dans le tableau 2.

Partant du modèle de Heinzelman et *al.* ci-dessus, nous définissons la consommation énergétique totale, notée $\mathcal{E}_{totale}$, comme étant la quantité totale d'énergie consommée dans tout le réseau durant une période d'étude donnée.

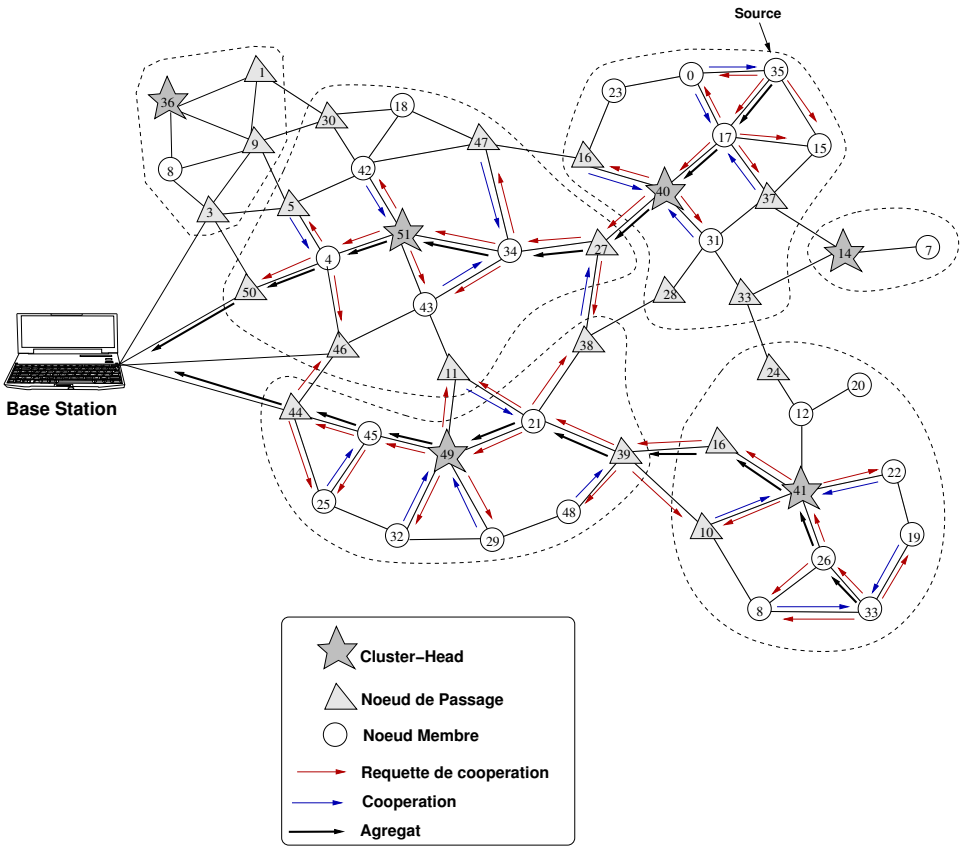


Figure 7 – Scénario de Routage avec Agrégation Totale (RAT)

Paramètre	définition	Unité
E_{elec}	Énergie pour faire marcher la radio	$50nJ/bit$
ε_{fs}	Modèle d'espace libre de l'amplificateur de l'émetteur	$10pJ/bit/m^2$
ε_{mp}	Modèle multi-path de l'amplificateur de l'émetteur	$0.0013\ pJ/bit/m^4$
l	Taille d'un message	2000 bits
d_0	Seuil de distance	$\sqrt{\varepsilon_{fs}/\varepsilon_{mp}}$

Tableau 2 – Paramètres du module radio utilisés

Cette consommation énergétique totale dépend fortement des communications c'est-à-dire des échanges de messages dans le réseau. En d'autres termes, plus il y a d'échanges de messages dans le réseau, plus la consommation énergétique est importante. En effet, comme la consommation énergétique pour les traitements CPU et les opérations de captage est très négligeable comparée à celle des communications (envois/réceptions de messages), elle est négligée dans le calcul de $\mathcal{E}_{totale}$ [3, 22, 29, 40, 46]. Ainsi, sur la base du

modèle énergétique de Heinzelman et *al.*, la consommation énergétique totale est obtenue est calculée à partir de l'équation 4 suivante.

$$\mathcal{E}_{totale} = \sum_{i=0}^{n-1} \mathcal{M}_i^{env} \times E_{Tx}(l,d) + \sum_{i=0}^{n-1} \mathcal{M}_i^{rec} \times E_{Rx}(l) \quad [4]$$

Dans l'équation 4, le terme $E_{Tx}(l,d)$ représente la quantité d'énergie consommée pour envoyer un message de longueur l bits sur une distance d exprimée en mètre. Cette quantité est calculée à partir de l'équation 2 de Heinzelman et *al.* ci-dessus. $E_{Rx}(l)$ est la quantité d'énergie consommée pour réceptionner un message de longueur l bits et elle est également calculée à partir de l'équation 3 de Heinzelman et *al.* ci-dessus. $\mathcal{M}_i^{env}$ représente le nombre total de messages envoyés et $\mathcal{M}_i^{rec}$ désigne le nombre total de messages reçus par un capteur i. n désigne le nombre total de nœuds dans le réseau.

6.1.2. Délai de bout en bout

Le délai de bout en bout est défini comme étant le temps d'acheminement de l'information depuis un capteur source jusqu'à la BS [8, 31, 30, 25]. Plusieurs facteurs influent sur ce délai. Nous avons le temps de propagation (δ_{prop}) qui est le temps nécessaire pour la transmission du message entre deux nœuds. Le temps de traitement (δ_{trait}) qui représente le temps nécessaire à un nœud pour réceptionner un message (démodulation), faire le traitement requis (récupérer l'adresse de destination et chercher la route dans le table de routage) puis retransmettre (modulation) vers la destination. Et le temps d'agrégation δ_{agreg} représente le temps d'attente pour fusionner les données. En fonction de ces différents facteurs et du nombre de liens de communication $\mathcal{N}$ qui est égal à $\alpha + 1$ avec α le nombre de nœuds traversés par un message, le délai de bout en bout pour chaque message délivré à la BS est calculé comme suit :

$$\Delta T = \mathcal{N} \times (\delta_{prop} + \delta_{trait} + \delta_{agreg}) \quad [5]$$

Dans le RSA, δ_{agreg} est nul car aucun nœud du réseau n'effectue d'agrégation. Pour le scénario de RAP, chaque *cluster-head* va attendre δ_{agreg} pour agréger les données de son *cluster*. Et pour le scénario de RAT, tout nœud attend un temps δ_{agreg} pour la coopération avec ses voisins. Les valeurs des paramètres δ_{prop}, δ_{trait} et δ_{agreg} seront données au niveau la Section 6.2. Dans chaque scénario, nous supposons que les temps de propagation, de traitement et d'agrégation sont identiques sur tous les nœuds réseau.

6.1.3. Durée de vie du réseau

La durée de vie du réseau est définie comme étant le *temps qui s'écoule* ou le *nombre de cycles de captages* depuis le déploiement des nœuds c'est-à-dire le démarrage de l'activité du réseau jusqu'au moment où le réseau devient non-fonctionnel [10, 14, 17, 39]. Le réseau est considéré comme non-fonctionnel selon les spécificités de l'application pour laquelle il a été déployé. Pour certaines applications critiques, le réseau est considéré comme non-fonctionnel dès la mort du premier nœud dans le réseau. Un nœud u est considéré comme mort dès l'épuisement de son énergie ($\mathcal{E}_u^{disp} = 0$). Dans ce cas, le nœud ne peut plus émettre ni réceptionner de messages. Pour d'autres applications, le réseau est considéré comme non-fonctionnel à partir de la mort d'un certain pourcentage

de nœuds. Le réseau peut également être considéré comme non-fonctionnel dès la perte de la connexité totale (perte de couverture) c'est-à-dire à partir du moment où la BS n'est plus atteignable depuis n'importe quel nœud du réseau.

Nous évaluons la durée de vie du réseau dans chacun des trois scénarios de routage (RSA, RAP et RAT) en mesurant d'abord le temps qui s'écoule et le nombre de cycles de captages depuis le déploiement des nœuds jusqu'à ce que le premier nœud disparait (($\mathcal{E}_u^{disp} = 0$)). Ensuite, sans refaire de *re-clustering*, nous évaluons la durée de vie maximale du réseau jusqu'à ce qu'à la perte de connexité totale. L'objectif est d'évaluer le nombre de nœuds i dont l'épuisement totale de la batterie conduit à une perte de connexité totale.

6.2. Environnement et paramètres de simulation

Dans cette section, nous présentons les différents paramètres de simulations utilisés. Nous résumons ces paramètres dans le tableau 3.

Nous considérons des topologies réseaux suivant une loi de Poisson [9, 12, 23, 37, 38]. Nous effectuons toutes nos mesures avec un intervalle de confiance de 99% [24]. Dans le but de considérer un RCSF hétérogène avec des nœuds dotés d'une faible capacité énergétique, nous attribuons à chaque nœud une valeur énergétique initiale prise aléatoirement à 1, 2 ou 3 joules [50, 35, 19, 20]. Pour étudier le passage à l'échelle, nous considérons des réseaux de taille allant de 100 à 1000 nœuds par pas de 100.

Pour le délai de propagation δ_{prop} et de traitement δ_{trait}, nous utilisons respectivement les valeurs 0.3 μs et 15,36 ms telles que définies dans la norme $802.15.4$ [21] et utilisées dans plusieurs travaux [32, 36, 48]. Et pour le délai d'agrégation δ_{agreg}, nous utilisons la valeur de 40 ms telle que utilisée dans la littérature [13, 43, 41].

Dans la stratégie de coopération des agents, nous attribuons équitablement un poids de $0,25$ à chacun des critères pour le calcul du coefficient de pertinence de la coopération. Pour favoriser un ou plusieurs critère (s) au détriment des autres, il est possible de jouer sur la valeur de leurs poids. Nous fixons le seuil de pertinence $\mathcal{R}$ (cf. Section 4.2) dans l'intervalle $[0,5; 0,99]$. Dès qu'un agent calcule une valeur de pertinence comprise dans cette intervalle alors il décide de coopérer. Nous fixons cet intervalle assez large dans le but simuler le réseau pour favoriser plus d'échanges possibles et de simuler au mieux des conditions extrêmes de consommation énergétique. Pour cette même raison, nous fixons également l'intervalle de captage à une (1) seconde.

Nous supposons le scénario suivant dans nos simulations. Nous avons un réseau de capteurs où les nœuds sont équipés d'un capteur de température. Ainsi, durant la période d'observation considérée (temps de simulation), à toutes les secondes (cycle de captage), tous les nœuds recueillent la température environnant et l'envoie à destination de la BS.

6.3. Résultats de simulations

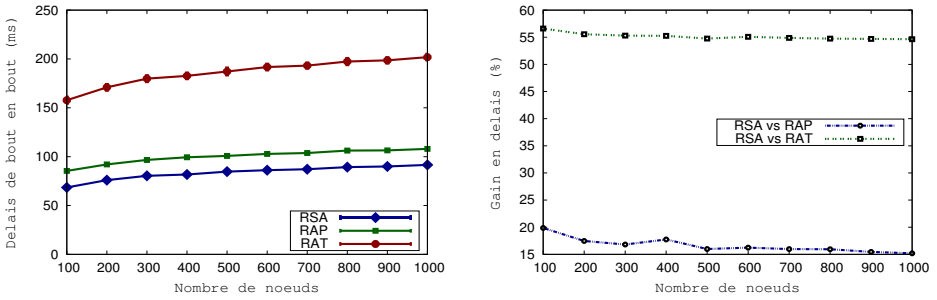
Dans cette section, nous présentons les résultats des évaluations des métriques du délai de bout en bout, de la consommation énergétique et de la durée de vie du réseau.

Tableau 3 – Paramètres de simulation pour l'évaluation des scénarios de routage

	Paramètres	Valeur
Constantes	Taille d'un message	2000 bits
	Portée radio	100 mètres
	Modèle de graphe	Distribution de Poisson
	Nombre de simulations pour chaque taille du réseau	100
	Intervalle de confiance	99%
	Paramètre k	2
	Degré moyen du réseau	6
	Intervalle de captage τ_s	1 s
	Délais de propagation δ_{prop}	0.3 μ_s
	Délais de traitement δ_{trait}	15,36 ms
	Délais d'agrégation δ_{agreg}	40 ms
	Poids pour la stratégie des agents (w_e, w_p, w_d, w_i)	$0,25$
Variables	Énergie initiale	{1,2,3} Joules
	Seuil de Pertinence $\mathcal{R}$	$[0,5; 0,99]$
	Nombre de nœuds	[100,1000]
	Temps de simulation $\mathcal{T}$	$1000s$, 10 jours

6.3.1. Délai de bout en bout

Nous entamons les évaluations de performances du RSA, du RAP et du RAT en mesurant leurs délais de bout en bout. Avec les résultats illustrés au niveau de la figure 8, nous considérons un temps de simulation fixé à 1000 secondes et un réseau de taille allant de 100 à 1000 nœuds.



(a) Délais de bout en bout de RSA, RAP et RAT (b) Rapidité de RSA comparé au RAP et RAT

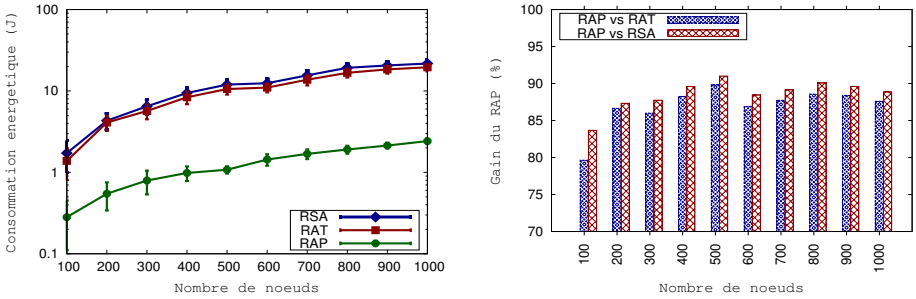
Figure 8 – Délai de bout en bout

Nous observons une évolution logarithmique des délais de bout en bout en fonction de la taille du réseau. Cela montre que les trois scénarios permettent le passage à l'échelle. Puis, nous remarquons que le RSA présente un meilleur délai suivit du scénario de RAP ensuite arrive en dernière position le RAT. Plus précisément, la figure 8(b) montre que le RSA est plus rapide de 15.18% à 19.85% par rapport au RAP et de 54.64% à 56.61% par

rapport au RAT. En effet, dans le RAS, il y a aucune attente. Les données collectées sont routées aussitôt, de proche en proche, jusqu'à la BS via les *cluster-heads*. Alors que dans le RAP, chaque *cluster-head* attend un temps δ_{agreg} pour procéder à la fusion de toutes les données de son *cluster*. Dans le cas du RAT, à chaque fois qu'un nœud collecte une donnée, une session coopération est initiée avec les nœuds du voisinage. Cela nécessite une attente afin de fusionner toutes les données venant éventuellement du voisinage qui ont accepté de coopérer. Comme ce procédé se répète tout au long du chemin jusqu'à la BS, le RAT est donc 54.64% à 56.61% plus lent que le RSA.

6.3.2. Consommation énergétique

En reprenant les expériences présentées précédemment, nous mesurons les consommations énergétiques des trois scénarios (cf. figure 9). Nous constatons qu'elles suivent une évolution logarithmique avec l'augmentation de la taille du réseau ; ce qui montre le passage à l'échelle.



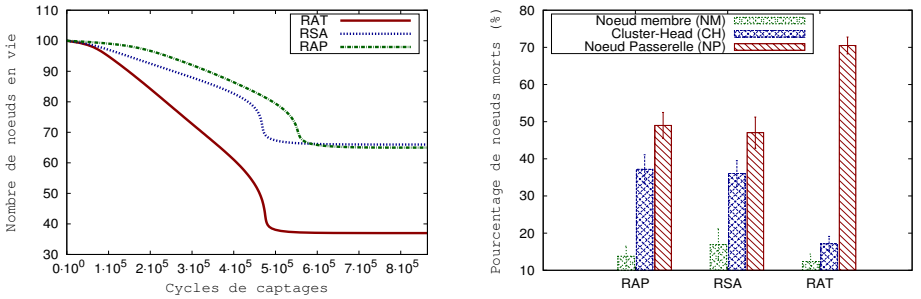
(a) RSA - RAP - RAT (b) RAP vs RSA - RAP vs RAT

Figure 9 – Consommation énergétique totale

Nous observons que le RAP consomme moins d'énergie par rapport aux deux autres scénarios. Plus précisément, La figure 9(b) montre que le RAP réduit l'énergie consommée dans le réseau de 83.67% à 91.01% par rapport au scénario de RSA et de 79.64% à 89.80% par rapport au RAT. L'avantage du RAP est que, comme durant chaque cycle de captages, un seul message contenant toutes les données collectées au sein de chaque *cluster* est envoyé par le *cluster-head* à destination de la BS, la consommation énergétique résultante en est considérablement réduite. Le RSA engendre plus de trafic car toute donnée captée est aussitôt acheminée vers la BS. Logiquement, cela augmente le trafic dans le réseau et entraine une consommation plus élevée en énergie.

6.3.3. Durée de vie du réseau

Pour évaluer la durée de vie du réseau, nous considérons le cas suivant. Nous partons d'un réseau structuré en *clusters* puis nous exécutons les trois scénarios de routage. Nous évaluons d'abord le temps auquel le premier nœud disparait du réseau du fait son épuisement énergétique. Puis, nous cherchons la durée de vie maximale du réseau avant qu'un *re-clustering*, causé par une perte de connexité totale, soit nécessaire. Nous considérons un réseau de taille fixée à 100 nœuds comme dans [22] et à toutes les secondes, chaque capteur collecte une donnée et l'envoie à destination de la BS. La figure 10(a) montre la durée vie des différents scénarios de routage.



(a) Durée de vie en cycles de captages (b) Pourcentage nœuds morts en fonction des statuts

Figure 10 – Durée de vie du réseau

Intéressons nous d'abord au temps qui s'écoule jusqu'à ce que le premier nœud disparait du réseau pour chaque scénario. Le tableau 4 donne ce temps pour chaque scénario. Comme nous avons fixé une période de captages à 1 seconde afin de stimuler le réseau au maximum, le premier nœud disparait après 41h56min (151018 cycles) dans le scénario du RAP, 17h41min dans le RSA et 15h42min dans le RAT. Nous constatons ainsi que le premier nœud disparait plus rapidement dans le RAT puis le RSA et le RAP. Ceci est causé par les échanges de messages nécessaires pour la coopération entre les agents dans le RAT. Donc, en considérant que le réseau est non-fonctionnel dès la disparition du premier nœud, il serait nécessaire d'effectuer un *re-clustering* à ces instants.

	Temps de disparition du premier nœud
RAP	41h 56min 57s
RSA	17h 41min 44s
RAT	15h 42min 57s

Tableau 4 – Temps de disparition du premier nœud dans le réseau

Supposons que le réseau reste toujours fonctionnel afin d'évaluer la durée de vie maximale que nous pouvons avoir jusqu'à la perte de connexité totale. Celle-ci est matérialisée par les courbes qui restent constantes et ne décroissent plus. Nous constatons qu'elle intervient dès la disparition d'environ 35% des nœuds du réseau dans le RAP et RSA. Dans le RAT, elle est observée à partir de la disparition d'environ 63% de nœuds du réseau. En effet, d'après la figure 10(b), la perte de connexité totale dans le RAP et le RSA est causée par la disparition d'environ 37% des *cluster-heads* du réseau. Dans le RAT, environ 17% des *cluster-heads* ont disparu à la perte de connexité totale. Nous obtenons donc la durée de vie maximale dans le RAT avant qu'un *re-clustering* ne soit nécessaire. En effet, dans le RAT, les *cluster-heads* qui coordonnent les communications sont ménagés au détriment des autres nœuds qui travaillent le plus. Nous pouvons également noter que dans le RAP,

l'agrégation partielle au niveau des *cluster-heads* réduit les messages qui transitent dans le réseau. Donc, il prolonge la durée de maximale comparé au RSA.

7. Conclusion et perspectives

Dans cet article, nous avons mené une étude complète visant à proposer trois stratégies de routage, intégrant différents niveau d'agrégation, afin d'acheminer les données collectées dans les réseaux de capteurs sans fil structurés en *clusters* auto-stabilisants. Ces trois stratégies sont les suivantes : *(i)* le Routage Sans Agrégation (RSA), *(ii)* le Routage avec Agrégation Partielle et *(iii)* le Routage avec Agrégation totale (RAT). Pour ce faire, nous nous sommes fondé sur notre schéma de *clustering* auto-stabilisant et y avons intégré un mécanisme intelligent d'agrégation de données à base de système d'agents coopératifs.

Pour valider les approches de routage proposées, nous avons évalué et comparé leurs performances en mesurant le délai de bout en bout, la consommation énergétique et la durée de vie du réseau. Les résultats de simulation sous OMNeT++ ont montré que le RSA minimise les délais de communication, le RAP réduit la consommation énergétique totale et le RAT prolonge la durée de vie des *cluster-heads*.

Dans nos futurs travaux, comptons orienter nos recherches vers deux axes. *(i)* Comparer les approches que nous mis en œuvre avec d'autres solutions de la littérature. *(ii)* Mener une étude de cas de nos solutions dans la plateforme de réseaux de capteurs *IoT-LAB* [4].

8. Bibliographie

[1] K. Akkaya, M. Younis, and M. Youssef. Efficient Aggregation of Delay-Constrained Data in Wireless Sensor Networks. In *Proceedings of the ACS/IEEE International Conference on Computer Systems and Applications*, AICCSA'05, pages 904–909, 2005.

[2] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks : a survey. *Computer Networks : The International Journal of Computer and Telecommunications Networking*, 38(4) :393–422, 2002.

[3] G. Anastasi, M. Conti, M. Di Francesco, and A. Passarella. Energy Conservation in Wireless Sensor Networks : A Survey. *Ad Hoc Networks*, 7(3) :537–568, 2009.

[4] Mandicou Ba, Olivier Flauzac, Bachar Salim Haggar, Florent Nolot, and Ibrahima Niang. Self-Stabilizing k-hops Clustering Algorithm for Wireless Ad Hoc Networks. In *Proceedings of the 7th ACM International Conference on Ubiquitous Information Management and Communication*, IMCOM'13, pages 38 :1–38 :10, Kota Kinabalu, Malaysia, 2013. **Best Paper Award**.

[5] Mandicou Ba, Olivier Flauzac, Rafik Makhloufi, Florent Nolot, and Ibrahima Niang. Comparison Between Self-Stabilizing Clustering Algorithms in Message-Passing

4. https ://www.iot-lab.info

Model. In *Proceedings of the 9th International Conference on Autonomic and Autonomous Systems*, ICAS'13, pages 27–32, 2013.

[6] Mandicou Ba, Olivier Flauzac, Rafik Makhloufi, Florent Nolot, and Ibrahima Niang. Evaluation Study of Self-Stabilizing Cluster-Head Election Criteria in WSNs. In *Proceedings of the 6th International Conference on Communication Theory, Reliability, and Quality of Service*, CTRQ'13, pages 64–69, 2013. **Best Paper Award**.

[7] Mandicou Ba, Olivier Flauzac, Rafik Makhloufi, Florent Nolot, and Ibrahima Niang. Fault-Tolerant and Energy-Efficient Generic Clustering Protocol for Heterogeneous WSNs. *International Journal On Advances in Networks and Services*, 6(3-5) :231–245, 2013.

[8] S. Balamurugan, S. Saraswathi, and A. Mullaivendhan. Delay Sensitive Optimal Anycast Technique to Maximize Lifetime in Asynchronous WSN's. *Procedia Engineering*, 38(0) :3351–3361, 2012.

[9] S. Bandyopadhyay and E. J. Coyle. An Energy Efficient Hierarchical Clustering Algorithm for Wireless Sensor Networks. In *Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, INFOCOM'03, pages 1713–1723, 2003.

[10] M. Bhardwaj, T. Garnett, and A. P. Chandrakasan. Upper Bounds on the Lifetime of Sensor Networks. In *Proceedings of the IEEE International Conference on Communications*, ICC'01, pages 785–790, 2001.

[11] E. Caron, A. K. Datta, B. Depardon, and L. L. Larmore. A Self-Stabilizing k-Clustering Algorithm for Weighted Graphs. *Journal of Parallel and Distributed Computing*, 70(11) :1159–1173, 2010.

[12] J. Chen, C. S Kim, and S. Fu. A Distributed Clustering Algorithm for Voronoi Cell-Based Large Scale Wireless Sensor Network. In *Proceedings of the International Conference on Communications and Mobile Computing*, CMC'10, pages 209–213, 2010.

[13] M. Chen, T. Kwon, Y. Yuan, and V. Leung. Mobile Agent Based Wireless Sensor Networks. *Journal of Computers*, 1(1), 2006.

[14] Y. Chen and Q. Zhao. On The Lifetime of Wireless Sensor Networks. *IEEE Communications Letters*, 9(11) :976–978, 2005.

[15] A. K. Datta, L. L. Larmore, and P. Vemula. A Self-Stabilizing O(k)-Time k-Clustering Algorithm. *The Computer Journal*, 53(3) :342–350, 2010.

[16] A.K. Datta, S. Devismes, and L. L. Larmore. A Self-Stabilizing O(n)-Round k-Clustering Algorithm. In *Proceedings of the 28th IEEE International Symposium on Reliable Distributed Systems*, SRDS'09, pages 147–155, 2009.

[17] I. Dietrich and F. Dressler. On the Lifetime of Wireless Sensor Networks. *ACM Transactions on Sensor Networks*, 5(1) :5 :1–5 :39, 2009.

[18] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi. In-Network Aggregation Techniques for Wireless Sensor Networks : a survey. *IEEE Wireless Communications*, 14(2) :70–87, 2007.

[19] T. Gao, R. Jin, J. Song, T. Xu, and L. Wang. Energy-Efficient Cluster Head Selection Scheme Based on Multiple Criteria Decision Making for Wireless Sensor Networks. *Wireless Personal Communications*, 63(4) :871–894, 2012.

[20] D. Gong, Y. Yang, and Z. Pan. Energy-Efficient Clustering in Lossy Wireless Sensor Networks. *Journal of Parallel and Distributed Computing*, 73(9) :1323–1336, 2013.

[21] IEEE Working Group. 802.15 WPAN Task Group 4 (TG4). http ://www.ieee802.org/15/pub/TG4.html, 2014.

[22] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan. Energy-Efficient Communication Protocol for Wireless Microsensor Networks. In *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, HICSS '00, pages 8020–8013, 2000.

[23] J. C. Hou, N. Li, and I. Stojmenovic. *Topology Construction and Maintenance in Wireless Sensor Networks*, pages 311–341. 2005.

[24] R. Jain. *The Art of Computer Systems Performance Analysis : Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, 1991.

[25] B Jiang, K. Han, B. Ravindran, and H. Cho. Energy Efficient Sleep Scheduling Based on Moving Directions in Target Tracking Sensor Network. In *Proceedings of the 6th IEEE International Symposium on Parallel and Distributed Processing*, IPDPS'08, pages 1–10, 2008.

[26] C. Johnen and F. Mekhaldi. Self-Stabilization versus Robust Self-Stabilization for Clustering in Ad-Hoc Network. In *Proceedings of the 17th international conference on Parallel processing - Volume Part I*, Euro-Par'11, pages 117–129, 2011.

[27] C. Johnen and L. Nguyen. Robust Self-Stabilizing Weight-Based Clustering Algorithm. *Theoretical Computer Science*, 410(6-7) :581–594, 2009.

[28] D. B. Johnson, D. A. Maltz, and J. Broch. Ad hoc networking. chapter DSR : The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks, pages 139–172. 2001.

[29] A. Karahan, I. Erturk, S. Atmaca, and S. Cakici. Effects of Transmit-Based and Receive-Based Slot Allocation Strategies on Energy Efficiency in WSN MACs. *Ad Hoc Networks*, 13(0) :404–413, 2014.

[30] J. Kim, X. Lin, and N. B. Shroff. Optimal Anycast Technique for Delay-sensitive Energy-constrained Asynchronous Sensor Networks. *IEEE/ACM Transactions on Networking*, 19(2) :484–497, 2011.

[31] J. Kim, X. Lin, N. B. Shroff, and P. Sinha. Minimizing Delay and Maximizing Lifetime for Wireless Sensor Networks with Anycast. *IEEE/ACM Transactions on Networking*, 18(2) :515–528, 2010.

[32] A. Koubaa, M. Alves, and E. Tovar. GTS Allocation Analysis in IEEE 802.15.4 for Real-Time Wireless Sensor Networks. In *Proceedings of the 20th international conference on Parallel and distributed processing*, IPDPS'06, pages 176–176, 2006.

[33] J. Kuroiwa, Y. Yamauchi, Weihua Sun, and M. Ito. A Self-Stabilizing Algorithm for Stable Clustering in Mobile Ad-Hoc Networks. In *Proceedings of the 4th IFIP International Conference on New Technologies, Mobility and Security*, NTMS'11, pages 1–7, 2011.

[34] A. Larsson and P. Tsigas. A Self-stabilizing (k,r)-clustering Algorithm with Multiple Paths for Wireless Ad-hoc Networks. In *Proceedings of the 31st International Conference on Distributed Computing Systems*, ICDCS'11, pages 353–362, 2011.

[35] Z. Liu, Q. Zheng, L. Xue, and X. Guan. A Distributed Energy-Efficient Clustering Algorithm With Improved Coverage in Wireless Sensor Networks. *Future Generation Computer Systems*, 28(5) :780–790, 2012.

[36] F. Meng and Y. Han. A New Association Scheme of IEEE 802.15.4 for Real-Time Applications. In *Proceedings of the 5th International Conference on Wireless communications, networking and mobile computing*, WiCom'09, pages 3432–3436, 2009.

[37] N. Mitton, E. Fleury, I. Guerin Lassous, and S. Tixeuil. Self-Stabilization in Self-Organized Multihop Wireless Networks. In *25th IEEE International Conference on Distributed Computing Systems Workshops*, ICDCSW '05, pages 909–915, 2005.

[38] M. Nasim, S. Qaisar, and S. Lee. An Energy Efficient Cooperative Hierarchical MIMO Clustering Scheme for Wireless Sensor Networks. *Sensors*, 12(1) :92–114, 2011.

[39] T. V. Padmavathy and M. Chitra. Extending the Network Lifetime of Wireless Sensor Networks Using Residual Energy Extraction - Hybrid Scheduling Algorithm. *International Journal of Communications, Network and System Sciences*, 3(1) :98–106, 2010.

[40] G. J. Pottie and W. J. Kaiser. Wireless Integrated Network Sensors. *Communications of the ACM*, 43(5) :51–58, 2000.

[41] A. Sardouk. *Agrégation de donnée dans les réseaux de capteurs sans fil à base d'agents coopératifs*. PhD thesis, 2010.

[42] A. Sardouk, M. Mansouri, L. Merghem-Boulahia, D. Gaïti, and R. Rahim-Amoud. Multi-Agent System Based Wireless Sensor Network for Crisis Management. In *IEEE Global Telecommunications Conference*, GLOBECOM'10, pages 1–6, 2010.

[43] A. Sardouk, R. Rahim-Amoud, L. Merghem-Boulahia, and D. Gaïti. A Multi-criterion Data Aggregation Scheme for WSN. In *Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*, WIMOB '09, pages 30–35, 2009.

[44] A. Sardouk, R. Rahim-Amoud, L. Merghem-Boulahia, and D. Gaïti. Data Aggregation Scheme for a Multi-Application WSN. In *Wired-Wireless Multimedia Networks and Services Management*, volume 5842 of *Lecture Notes in Computer Science*, pages 183–188. 2009.

[45] A. Sardouk, R. Rahim-Amoud, L. Merghem-Boulahia, and D. Gaïti. Information-Importance Based Communication for Large-Scale WSN Data Processing. In *Wireless and Mobile Networking*, volume 308 of *IFIP Advances in Information and Communication Technology*, pages 297–308. 2009.

[46] K. SOHRABY, M. DANIEL, and Z. TAIEB. *Wireless Sensor Networks Technology Protocols and Applications*. Wiley, 2007.

[47] Y. F. Wen, T. A. F. Anderson, and D. M. W. Powers. On Energy-Efficient Aggregation Routing and Scheduling in IEEE 802.15.4-Based Wireless Sensor Networks. *Wireless Communications and Mobile Computing*, 14(2) :232–253, 2014.

[48] K. Wook. Short Clear Channel Assessment in Slotted IEEE 802.15.4 Networks. *Wireless Personal Communications*, 71(1) :735–744, 2013.

[49] H. Xiaofeng, Xiang C., E.L. Lloyd, and Chien-Chung S. Fault-tolerant relay node placement in heterogeneous wireless sensor networks. *IEEE Transactions on Mobile Computin,*, 9(5) :643–656, 2010.

[50] Z. Yong and Q. Pei. A Energy-Efficient Clustering Routing Algorithm Based on Distance and Residual Energy for Wireless Sensor Networks. *Procedia Technology*, 29(0) :1882–1888, 2012.